International Journal of Research in Computer Science
eISSN 2249-8265 Volume 2 Issue 3 (2012) pp. 17-23

www.ijorcs.org

A RANDOMIZED LOAD BALANCING ALGORITHM IN GRID USING MAX MIN PSO ALGORITHM

C.Kalpana[1], U.Karthick Kumar[2], R.Gogulan[3]

[1,2]*Assistant Professor, Department of Computer Science, V.L.B Janakiammal College of Arts and Science, Coimbatore, Tamilnadu, India*
Email: miss.c.kalpana@gmail.com, u.karthickkumar@gmail.com

[3]*Research Scholar, School of Computer Sciences, Bharath University, Selaiyur, Chennai*
Email: r.gogul@lycos.com

Abstract: *Grid computing is a new paradigm for next generation computing, it enables the sharing and selection of geographically distributed heterogeneous resources for solving large scale problems in science and engineering. Grid computing does require special software that is unique to the computing project for which the grid is being used. In this paper the proposed algorithm namely dynamic load balancing algorithm is created for job scheduling in Grid computing. Particle Swarm Intelligence (PSO) is the latest evolutionary optimization techniques in Swarm Intelligence. It has the better performance of global searching and has been successfully applied to many areas. The performance measure used for scheduling is done by Quality of service (QoS) such as makespan, cost and deadline. Max PSO and Min PSO algorithm has been partially integrated with PSO and finally load on the resources has been balanced.*

Keywords: *Component, Computational Grid, grid Scheduling, Load balancing, Swarm Intelligence.*

I. INTRODUCTION

Grid scheduling is a sophisticated decision making that operates at different levels of grids. Local scheduling is used at the level of cluster, usually to balance load. Global schedulers that is grid schedulers may be used to map user jobs to resource according to their requirements [1].

Computational Grid: Grid computing is a network that may connect many computers through connection to solve problems requiring a large number of processing cycles and involving huge amount of data. Most commonly grid network is used to take a processor speed when the computers are not in use like screen saver mode. In that time that particular system processor speed is equally passes to all other working computers. So the entire network speed will be high. Grids integrate network, computation, information, and communication to providing a virtual environment for task management. The grid network can be typically grouped with an autonomous administrative domain, communicate with high-speed processor. The more time it takes for an ant to travel down the path and back again, the more time the pheromones have to evaporate [2]. A short path, by comparison, gets marched over more frequently, and thus the pheromone density becomes higher on shorter paths than longer ones. Pheromone evaporation also has the advantage of avoiding the convergence to a locally optimal solution. If there were no evaporation at all, the paths chosen by the first ants would tend to be excessively attractive to the following ones. In that case, the exploration of the solution space would be constrained [3].

Scheduling Grid: A High level scheduler can be used to select brokers or grids to a specific job. The scheduling mechanisms are related to the adaption decisions in autonomously behaving systems. A Grid scheduler computes one or more schedules for jobs that can be specified at launching-time. The structure of a scheduler may depend on the number of resources managed, and the domain in which resources are located [4]. Basically there are three different schedulers are used in grid network; centralized, decentralized and hierarchical. Grid scheduling algorithms are classified into two categories: Static and Dynamic. Launching-time scheduling for parallel process applications, where configuration and mapping decision are taken before execute the application.

PSO is created for task scheduling. And some the properties are used to update the particle position and find the current position of each particle. These are,

(i) Exchange information with its neighbors
(ii) Memorize a previous position
(iii) Ability to use information to make a decision.

The velocity updation is based on three terms; Inertia, Cognitive Learning, Social Learning. These are used for particle's migration. The rest of the paper is organized as follows: Section 2 presents the Related Work, Section 3 describes the Problem Formulation. In section 4 proposed work of PSO. Section 5 explained the Max Min PSO. Section 6 done the Randomized Load balancing. Results are discussed in section 7. Section 8 gives the Conclusion of this paper and finally future work is done in section 9.

II. RELATED WORK

Natural behavior of ants had inspired scientists to mimic insect operational methods to solve real-life complex problems. By observing ant behavior, scientists have begun to understand their means of communication. Ants secrete pheromone while traveling from the nest to food, and vice versa in order to communicate with one another to find the shortest path. K. Kousalya and P. Balasubramanie used task parceling based Ant Algorithm for Grid Scheduling [8]. Abraham et al. used Tabu Search and Simulated Annealing for scheduling jobs on computational grids [9]. An improved PSO Algorithm was proposed by BU Yan-ping, et al., against the optimal objective to minimize the total completion time [10]. This presents an improved particle swarm optimization (PSO) algorithm with discrete coding rule for grid scheduling problem. The collective and social behavior of living creatures motivated researchers to undertake the study of Swarm Intelligence. The main properties of the collective behavior can be pointed out as follows.

The idea of PSO originates from the social behavior of swarms of birds or fish schools. It has been reported that each bird or fish in the swarm in a school moves according to the information given by others in the neighborhood. Similar to the GA, there is a population called a swarm in PSO, and every individual in the swarm is called a particle which represents a solution to the problem. In an iteration, each particle moves in the search space from one position to another according to the information from its neighborhood.

III. PROBLEM FORMULATION

The idea of PSO originates from the social behavior of swarms of birds or fish schools. It has been reported that each bird or fish in the swarm in a school moves according to the information given by others in the neighborhood. Similar to the GA, there is a population called a swarm in PSO, and every individual in the swarm is called a particle which represents a solution to the problem. In an iteration each particle moves in the search space from one position to another according to the information from its neighborhood. The distance for which the particle moves in one iteration is called velocity and the position of the particle is evaluated by an object function. Each particle records the best position it found in the previous iteration as a local best. A global best position is chosen from the local bests of all the particles.

A. PSO Algorithm

1. Initialize a population array of particles with random positions and velocities on *D* dimensions in the search space.

   loop

2. For each particle, evaluate the desired optimization fitness function in *D* variables.
3. Compare particle's fitness evaluation with its $pbest_i$. If current value is better than $pbest_i$, then set $pbest_i$ equal to the current value.
4. Identify the particle in the neighborhood with the best success so far, and assign its index to the variable *gbest*
5. Change the velocity and position of the particle according to the velocity equation
6. If a criterion is met (usually a sufficiently good fitness or a maximum number of iterations), exit loop.

   end loop

B. Position Updation

Based on velocity particle position is updated. From the Updation pbest and gbest [14] are calculated according to QoS constraints.

$$V_i^{k+1} = \omega V_i^k + c_1 rand_1 x(pbest_i - X_i^k) + c_2 rand_2 x(gbest_i - X_i^k) \quad (1)$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \quad (2)$$

Where
V_i^k = Velocity of particle i at iteration k
V_i^{k+1} = Velocity of particle i at iteration k+1
ω = Inertia Weight
c_j = Acceleration Coefficients j=1,2
$rand_i$ = Random number between 0 and 1
X_i^k = Current position of particle i at iteration k
$pbest_i$ = Best position of particle i
gbest = Position of best particle in a population
X_i^{k+1} = Position of the particle I at iteration k+1.

IV. PROPOSED PSO

The main objective of this paper is to reduce the time of the schedule required to complete the execution of all tasks by a set of resources. Also the tasks are rescheduled from over loaded resources to under loaded resources which results in further reduction of makespan. The makespan, cost and deadline are used as performance measure. Particles are constructed by fitness value based on three QoS constraints such as makespan, cost and deadline.

A. Makespan Constraints

The particle with the minimum time has the greater probability to be chosen. Initialization of particle with makespan as

$$X_i^{k+1} = t_j \, x \, p_j \quad (3)$$

$Pbest_i = maximum(X_i^k)$
$Gbest = maximum(Pbest)$
Where t_i = Time of task,
P_j = Processor speed of resources.

B. Cost Constraints

The particle with the minimal cost has the greater probability to be chosen. Initialization of particle with cost as

$$X_i^{k+1} = maximum\ (Cost_j) \quad (4)$$

Pbest$_i$ = *maximum* (X_i^k)
Gbest = maximum (Pbest)
Where Cost$_j$ = Cost of each task.

C. Deadline Constraints

The particle with the minimal deadline has the greater probability to be chosen. Initialization of particle with deadline as

$$X_i^k = D_i - t_j \quad (5)$$

Where t_i = time of each task,
D_i =Deadline of each task.
X_i^k = Number of satisfied task with deadline

V. MAX-MIN PSO

In max pso tasks are scheduled in decreasing order by $t_1 > t_2 > t_3 \ldots t_N$, where N is the number of task. After the arrangement of task value velocity is calculated based on makespan. In max pso tasks are scheduled in increasing order by $t_1 < t_2 < t_3 \ldots t_N$, where N is the number of task. After the arrangement of task value velocity is calculated based on makespan. For load balancing, the one task from the overflowed resource is exchanged with a task from the under-flowed resource subject to the availability of the resources.

VI. RANDOMIZED LOAD BALANCING

Load balance in the computational grid is another important factor. Load balancing in order to share the available resources in computational grid, result in reduction of the average completion time of task. Load balance is applied here to interchange the maximum task value to minimum task value as well as minimum task value to maximum task value to the appropriate resources are balanced. This method can be iterated until all resources are balanced.

These techniques are implemented by the following formula,

$$R_l = Max\{\sum_{i=1}^{n} t_{kj}\}; l = 1,2,3, \ldots . M \quad (6)$$

R_l = Position of the maximum Resource
M = No. of Resource
N = No. of task allocated to particular resource

$$R_k = Min\{\sum_{i=1}^{n} t_{kj}\}; k = 1,2,3, \ldots . M \quad (7)$$

R_k = Position of the minimum resource
M = No. of Resource
N = No. of task allocated to particular resource
T_m = Interchange min task.
T_p = Interchange max task.
$T_m = R_k$ { Min (tn)}
$T_p = R_l$ { Max (tn)}

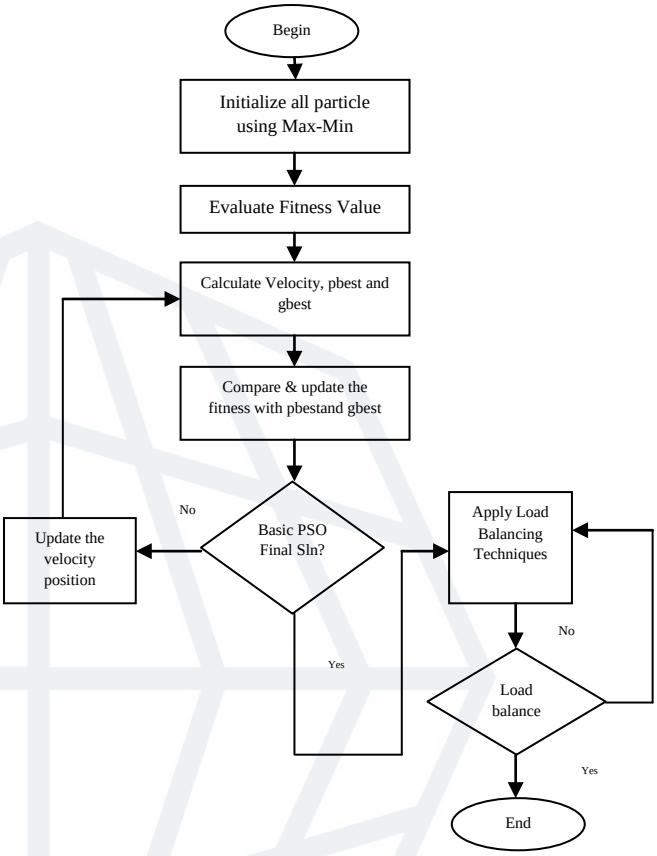

Figure 2. Flow chart

VII. RESULTS AND DISCUSSION

A Performance Comparison of basic PSO and Proposed PSO with ACO Here ACO and PSO algorithms are implemented in java. The comparison of these two algorithms is done on the basis of makespan, execution cost and deadline. The proposed and existing methods are simulated for various combinations of heterogeneity of task and resources with different Matrix such as 128 x 8, 256 x 16, 512 x 32, and 1024 x 64. Here, 128, 256, 512, 1024 represents number of tasks and 8, 16, 32, 64 represents number of resources. The performance of both existing and proposed method is evaluated by the makespan, cost and deadline. The existing and proposed method is tested using some range of ETC matrix. The result was compared with different QoS such as Makespan, Cost and Deadline. The existing and proposed method is tested using some range of ETC matrix. The range of tasks and resources are given below: PSO gives less makespan, cost and number of jobs completed within deadline is more compared to ACO. PSO approximately gives a reduction of 20% to 34% for makespan, 20% to 30% for cost and approximately 10% to 35% increase in Deadline compared to ACO in most of the cases.

Table 1: Task and Resource Range

Low Task Length and Low Heterogeneity Task Length	1000 – 2000
Low Task Length and High Heterogeneity Task Length	1000 – 5000
High Task Length and Low Heterogeneity Task Length	5001 – 7000
High Task Length and High Heterogeneity Task Length	5001 – 10000
Low Resource Speed and Low Heterogeneity Resource Speed	0 – 2
Low Resource Speed and High Heterogeneity Resource Speed	0 – 5
High Resource Speed and Low Heterogeneity Resource Speed	5 – 7
High Resource Speed and High Heterogeneity Resource Speed	5 – 10

Table 2: Low Task Length Low Resource Speed Low Hetrogenity Task Length Low Hetrogenity Resource Speed

Sche algo	Res Matx	Makespan(s)	Cost RS	Deadline (%)
ACO	128 x 8	70588.63	430679.48	58.50%
PSO		66968.27	411990.65	92.19%
ACO	256 x 16	70545.99	549916.91	64.06%
PSO		69737.25	496086.22	87.89%
ACO	512 x 32	77268.06	588891.78	70.12%
PSO		74406.38	524536.88	83.39%
ACO	1024 x 64	73386.33	560916.3	58.59%
PSO		71906.24	504461.49	81.25%

Table 3: Low Task Length Low Resource Speed Low Hetrogenity Task Length Low Hetrogenity Resource Speed for Basic PSO, Max PSO and Min PSO

Makespan for Basic PSO, Max PSO and Min PSO			
Resource Matrix	**PSO**	**MAX PSO**	**MIN PSO**
128 x 8	66968.27	59156.53	62144.37
256 x 16	69737.25	67503.64	69836.11
512 x 32	74406.38	68609.77	71217.66
1024 x 64	71906.24	69137.78	70965.34

Table 4: Low Task Length Low Resource Speed Low Hetrogenity Task Length Low Hetrogenity Resource Speed for Basic PSO with, Max PSO LB and Min PSO LB

MAKESPAN(S)				
Resource Matrix	**PSO**	**PSO LB**	**MAX PSO LB**	**MIN PSO LB**
128 x 8	66968.27	60034.6	57303.04	60061.96
256 x 16	69737.25	61911.82	58341.85	59828.23
512 x 32	74406.38	58696.6	56479.65	57989.22
1024 x64	71906.24	57081.44	56178.24	57367.23

Table 5 : Low Task Length Low Resource Speed Low Hetrogenity Task Length High Hetrogenity Resource Speed

Sche algo	Res Matx	Makespan(s)	Cost RS	Deadline (%)
ACO	128 x 8	175633.77	504420.98	78.91%
PSO		161918.29	477543.69	82.03%
ACO	256 x 16	144856.71	559932.47	58.98%
PSO		136406.67	479671.52	78.91%
ACO	512 x 32	152515.69	517781.59	68.16%
PSO		136557.76	479726.24	89.65%
ACO	1024 x 64	166365.29	526422.22	76.07%
PSO		148263.08	510066.58	90.04%

Table 6 Low Task Length Low Resource Speed Low Hetrogenity Task Length High Hetrogenity Resource Speed

Makespan for Existing PSO, Max PSO and Min PSO			
Resource Matrix	**PSO**	**MAX PSO**	**MIN PSO**
128 x 8	161918.29	105925.17	108953.77
256 x 16	136406.67	135169.43	139803.67
512 x 32	136557.76	131561.81	133358.47
1024 x 64	148263.08	136210.97	142015.31

Table 7 Low Task Length Low Resource Speed Low Hetrogenity Task Length High Hetrogenity Resource Speed for Load Balancing

MAKESPAN(S)				
Resource Matrix	**PSO**	**PSO LB**	**MAX PSO LB**	**MIN PSO LB**
128 x 8	161918.29	100339.28	97295	98924
256 x 16	136406.67	119926.85	118224	120569
512 x 32	136557.76	112595.53	108779	110326
1024x 64	148263.08	114001.38	112368	11696

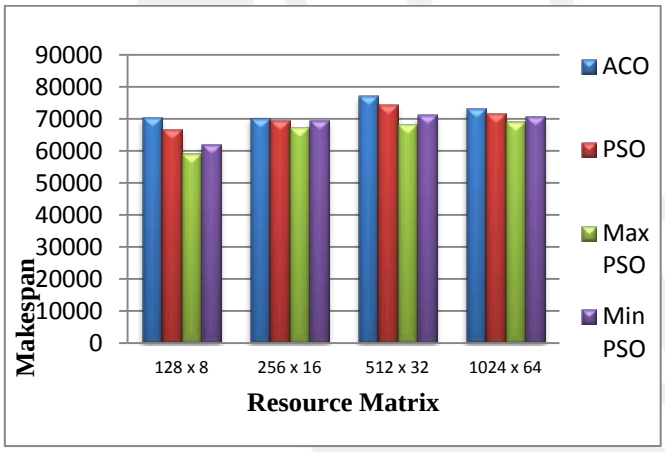

Figure 3. Performance Comparison of ACO with proposed PSO for LLLL type of ETC matrix for Makespan.

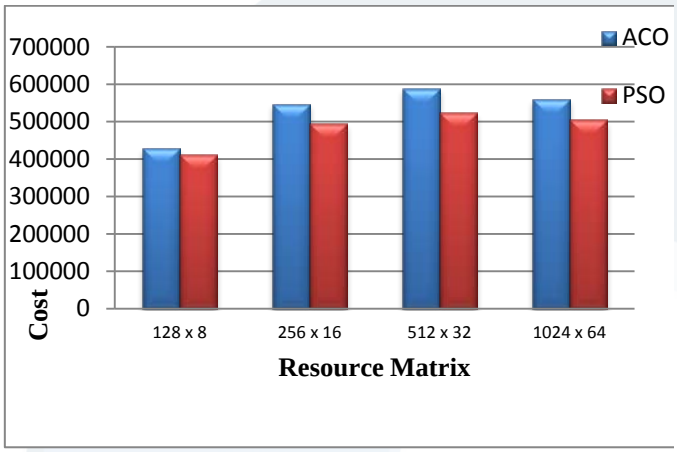

Figure 4. Performance Comparison of ACO with proposed PSO for LLLL type of ETC matrix for Cost.

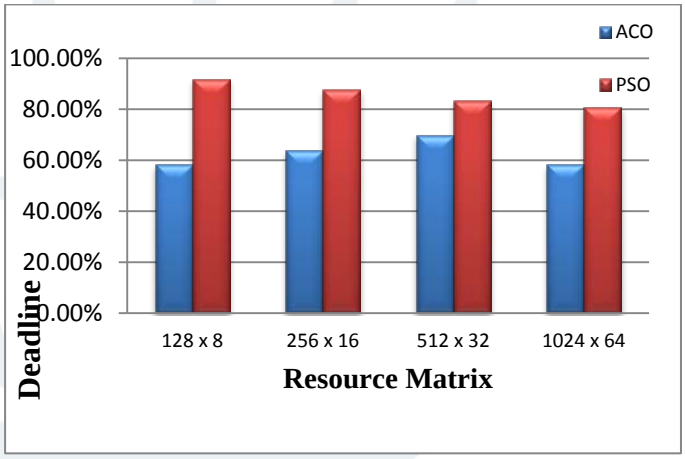

Figure 5. Performance Comparison of ACO with proposed PSO for LLLL type of ETC matrix for Deadline.

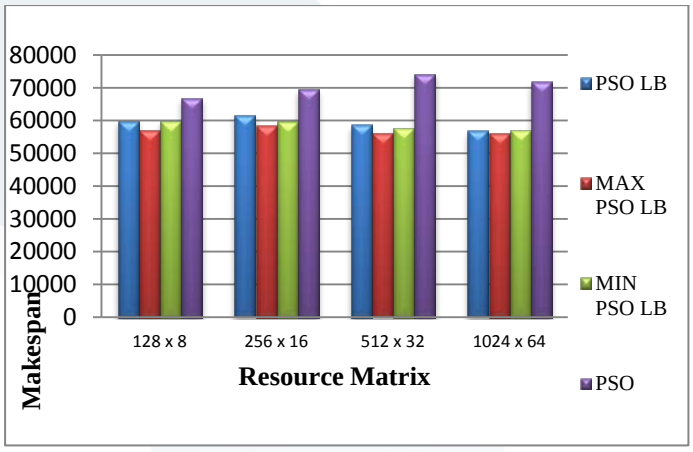

Figure 6. Performance Comparison of PSO, PSO LB, MAX PSO LB and MIN PSO LB for LLLL type of ETC matrix.

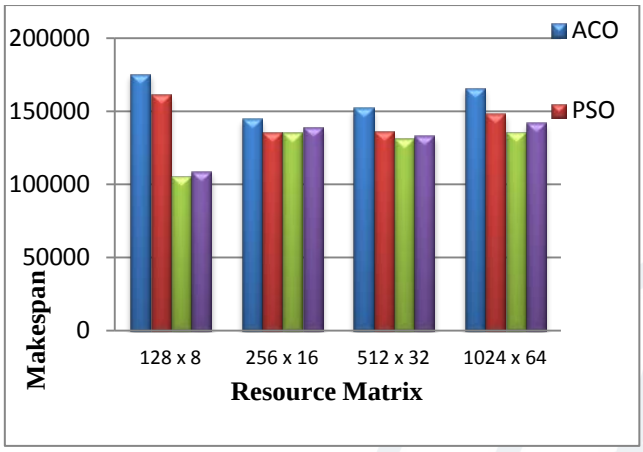

Figure 7. Performance Comparison of ACO with proposed PSO for LLLH type of ETC matrix for Makespan.

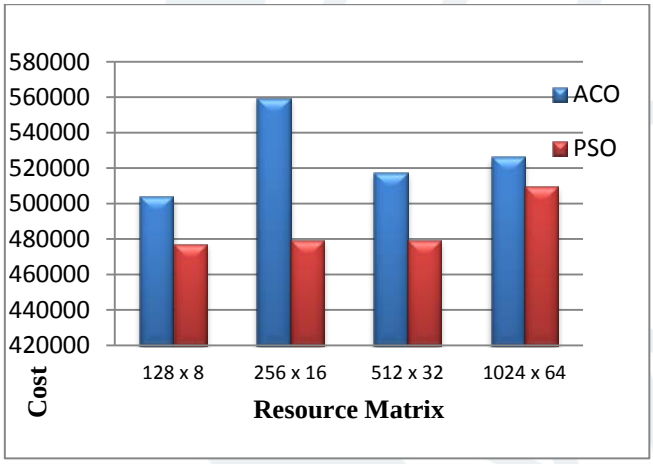

Figure 8. Performance Comparison of ACO with proposed PSO for LLLH type of ETC matrix for Cost

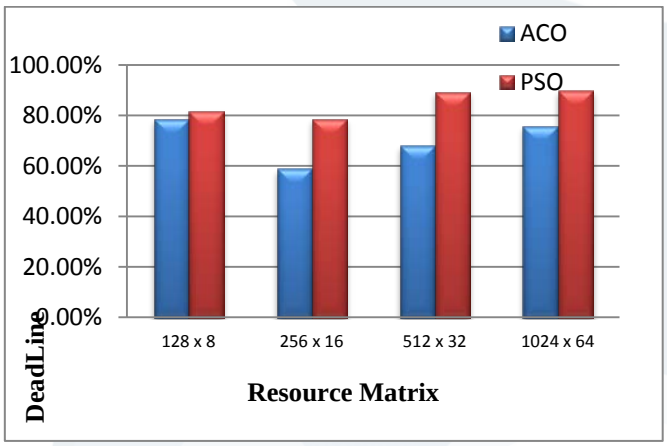

Figure 9. Performance Comparison of ACO with proposed PSO for LLLH type of ETC matrix for Deadline.

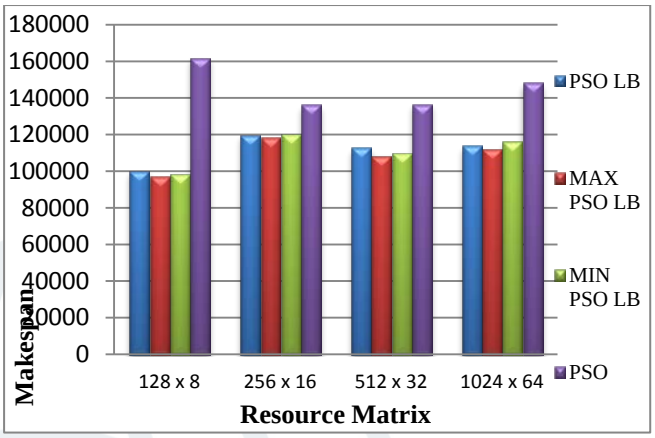

Figure 10. Performance Comparison of PSO, PSO LB, MAX PSO LB and MIN PSO LB for LLLH type of ETC matrix.

VIII. CONCLUSION

In this paper randomized load balancing, particle is calculated with max PSO and min PSO. This new concept is developed in java platform. The major objective of grid scheduling is to reduce the makespan, cost and increase the deadline. The algorithm is developed based on PSO to find a proper resource allocation to jobs in Grid Environment. In this paper, Particle Swarm Optimization (PSO) Algorithm is developed based on three QoS makespan, cost and Deadline. Results show that PSO is better than ACO according to makespan, cost and Deadline. finally the particle is randomly selected for load balancing. Future work can focus on, Reliability and throughput, To implement and evaluate different QoS and different Job error ratio.

IX. REFERENCES

Journals

[1] Dr.K.Vivekanandan, D.Ramyachitra, "A Study on Scheduling in Grid Environment",International Journal on Computer Science and Engineering (IJCSE), Vol. 3 No. 2 Feb 2011.

[2] K. Kousalya and P. Balasubramanie, "Task Severance and Task Parceling Based Ant Algorithm for Grid Scheduling", International journal of computational cognition (http://www.ijcc.us), vol. 7, no. 4, december 2009.

[3] Lei Zhang, et al, "A Task Scheduling Algorithm Based on PSO for Grid Computing", International Journal of Computational Intelligence Research, ISSN 0973-1873 Vol.4, 1 (2008), pp. 37–43.

[4] Wei-Neng Chen, Jun Zhang, "An Ant Colony Optimization Approach to a Grid workflow Scheduling Problem With Various QoS Requirements", IEEE Transactions on Systems man and cybernetics—Part c: Applications and Reviews, vol. 39, no. 1, January 2009.

[13] D. Merkle, M. Middendorf, and H. Schmeck, "Ant colony optimization for resource-constrained project scheduling", IEEE Trans. Evol. Comput.,vol. 6, no. 4, pp. 333–346, Aug. 2002.

[14] Pakize Erdogmus, "Particle swarm optimization performance on special linear programming problem", Scientific Research and Essays Vol. 5(12), pp. 1506-1518, 18 June, 2010.

[15] Grosu, D., Chronopoulos, A.T, "Noncooperative load balancing in distributed systems", Journal of Parallel Distrib.Comput. 65(9), 1022–1034 (2005).

Conference

[1] Karl Czajkowski, Steven Fitzgerald, Ian Foster, Carl Kesselman, "Grid Information Services for Distributed Resource Sharing", To appear in Proc. 10th IEEE Symp. on High Performance Distributed Computing, 2001.

[5] Alberto Colorni, Marco Dorigo, Vittorio Maniezzo, "Distributed Optimization by Ant Colonies", European Conference On Artificial Life, Paris, France, Elsevier Publishing, 134–142.

[7] J.Kennedy and R.Eberhart, "Particle swarm optimization", In Neural Networks, Proceedings, IEEEInternational Conference on, volume 4, pages 1942–1948 vol.4, 1995.

[9] Abraham, R. Buyya, and B. Nath, "Nature's heuristics for scheduling jobs on computational Grid", In The 8th IEEE International Conference on Advanced Computing and Communications (ADCOM 2000), India, 2000.

[11] Lei Zhang, Yuehui Chen, Bo Yang, "Task Scheduling Based on PSO Algorithm in Computational Grid", 2006 Proceedings of the 6th International Conference on Intelligent Systems Design and Applications, vol-2, 16-18 Oct, 2006, Jinan, China.

Books

[3] Bart Jacob, Michael Brown, Kentaro Fukui, Nihar Trivedi, "Introduction to Grid Computing", RedBook, December 2005.

Website

[6]http://en.wikipedia.org/wiki/Ant_colony_optimization_algorithms#cite_note-0

Other Articles

[2] D.P.Spooner, S.A.Jarvis, J.Cao, S.Saini and G.R.Nudd, "Local Grid Scheduling Techniques using Performance Prediction", High Performance Systems Group, Department of Computer Science, University of Warwick, Coventry, CV4 7AL, UK.

[16] Penmatsa, S., Chronopoulos, A.T, "Job allocation schemes in computational Grids based on cost optimization", In: Proceedings of 19th IEEE International Parallel and Distributed Processing Symposium,Denver, (2005).